

## CLAIMS

We claim:

5  
1. An extensible framework for use in a computer system that supports an object oriented programming environment and includes a memory in which data objects can be stored, the framework comprising a set of object classes that can be extended using object oriented principles to define an information handling  
10 application with data objects comprising a class of data source objects and a class of data consumer objects, and a mediation layer that defines an interface between the data source objects and the data consumer objects to permit data communications between the two data object classes, such that the class configuration of the data source objects can be specified independently of the class configuration of the data consumer objects.  
15

2. A computer system that supports an object oriented programming environment and includes access to data storage containing specifications for a set of object classes that can be extended using object oriented principles to define an  
20 information handling application with data objects comprising a class of data source objects and a class of data consumer objects, and a mediation layer that defines an interface between the data source objects and the data consumer objects to permit data information exchange between the two data object classes, such that the class configuration of the data source objects can be specified independently of the class  
25 configuration of the data consumer objects.

3. A computer system as defined in claim 2, wherein the mediation layer defines a data interface that provides an information exchange standard between the data source objects and the data consumer objects.

5 4. A computer system as defined in claim 3, wherein the data interface includes at least one attribute/metadata object class that specifies attributes, including metadata, and that provide declarative and procedural information relating to attributes in said data object.

10 5. A computer system as defined in claim 3, wherein the data interface includes at least one Relationship object class that specifies relationships between data source class objects.

15 6. A computer system as defined in claim 3, wherein the data interface includes at least one Domain Policy object class that specifies a group of related attributes, methods, and semantic information that indicates data processing to be available for the specified attributes, including the specific intent of said attributes.

20 7. A computer system as defined in claim 3, wherein the data interface includes at least one Event Change object class that specifies change event registration for detecting changes in the data objects.

25 8. A computer system as defined in claim 4, further including a TypeMetaData class of objects that specify semantic information indicating data processing to be available for specified attributes and intended use of the attributes of a data object.

9. A computer system as defined in claim 8, further including a TypeIO class of objects that define a desired format of a data object attribute specified by a TypeMetaData class.

5 10. A computer system as defined in claim 6, wherein an attribute alias may be specified by a user to indicate any data source attributes of an attribute domain that may be substituted with attributes of a different attribute domain.

10 11. A computer system as defined in claim 4, wherein said attribute/metadata interface provides input/output functions for display and editing of the attribute during runtime.

15 12. A computer system as defined in claim 4, further including a FieldMetaData attribute that specifies processing criteria for an attribute of a data object.

13. A computer system as defined in claim 4, wherein the processing criteria relates to a selected attribute of the data object for display processing.

20 14. A computer system as defined in claim 5, wherein the indicated relationship may be resolved using a version of the Relationship object in cache of the computer system.

25 15. A computer system as defined in claim 2, further including a collaboration facility that permits a user at a network computer remote from the computer system to share a view of a data object at the remote network computer.

16. A computer system as defined in claim 15, wherein the collaboration facility comprises a MetaView class of data objects that lightweight encapsulate elements of a data object for visual reconstruction on a display screen of the remote network computer.

17. A computer system as defined in claim 4, wherein the data interface provides an override function that can override default metadata of the data object attributes.

18. A computer system as defined in claim 6, wherein the data interface provides an override function that can override default metadata of the data object attributes.

19. A computer system as defined in claim 3, wherein data objects provided through the data interface comprise information that specifies data attributes, relationships, event broadcasting of changes in all registered data consumer objects, metadata for attributes that provide declarative and procedural information and/or input/output functions for display and editing, and related data items, and wherein the data objects can store references without direct access to a related data item.

20. A computer system as defined in claim 19, wherein the attributes and relationships include contextual metadata, and the events and methods are defined with respect to the contextual metadata.

21. A computer system as defined in claim 3, wherein the mediation layer further includes object classes that include one or more methods that provide data exposure, and further including a Data Source Interface object class of data objects

with methods that automatically determine which data exposure method will be used for data communications between the data source objects and data consumer objects.

22. A computer system as defined in claim 21, wherein the determined data exposure method comprises data object reflection and introspection.

23. A computer system as defined in claim 21, wherein the data exposure methods include a data source object method that provides a standard interface, a data source object and a Translator object that maps an alternate data interface to the standard interface, and a data source object that is inspected by the computer system to determine available data fields which are thereby exposed to data consumer objects by the standard interface.

24. A computer system as defined in claim 3, wherein the data interface of the mediation layer provides a wrapper for the data objects.

25. A computer system as defined in claim 24, wherein data exchange occurs without providing contextual data characteristics to a receiving data object.

26. A computer system as defined in claim 25, wherein the contextual data characteristics are supplied by the data object wrapper.

27. A computer system as defined in claim 3, wherein data exchange occurs without providing contextual data characteristics to a receiving data object.

28. A computer system as defined in claim 2, further including a plug-in manager object class that integrates service components into the application.

29. A computer system as defined in claim 28, wherein data service components interact with other components of the application to determine if a service is required and to determine parameters that are to be interchanged.

5

30. A computer system as defined in claim 29, wherein the service determination occurs upon introduction of newly loaded components.

31. A computer system as defined in claim 3, further including a translator facility that translates information concerning a data object into (1) attributes and metadata that define groups of data source and data consumer class attributes and (2) an object class that specifies at least one domain method for defining groups of data object attributes.

10

32. A computer system as defined in claim 24, further including an InfoModel data object class.

15

33. A computer system as defined in claim 27, wherein said contextual data characteristics are supplied by an InfoModel object.

20

34. A computer system as defined in claim 33, wherein the InfoModel object receives a data source object for exposure to the data consumer objects, selects available attributes and methods of the received data source object for data characterization, and determines if the addition of attributes or methods to the data source object are appropriate.

25

35. A computer system as defined in claim 6, wherein the domain is defined using XML schema.

36. A method of communicating data in a computer system that supports an object oriented programming environment and includes data storage or access to data storage containing data objects and specifications for a set of object classes that can be extended using object oriented principles to define an information handling application, the method comprising:

receiving one or more data objects comprising a class of data source objects;  
representing the data source objects in accordance with an Information Model class of objects of a mediation layer that defines a data interface between the data source objects and a class of data consumer objects;

wherein the Information Model objects include methods that permit data communications or information exchange between the two data object classes, such that the class configuration of the data source objects can be specified independently of the class configuration of the data consumer objects.

37. A method as defined in claim 36, further including specifying at least one attribute/metadata object class that specifies attributes, including metadata, and that provides declarative and procedural information relating to attributes in said data object.

38. A method as defined in claim 36, further including specifying a Relationship object class or classes that specify relationships between data source objects.

39. A method as defined in claim 36, further including specifying a Domain Policy or Domain object class or classes that specify a group of related attributes, methods, and semantic information that indicates data processing to be available for the specified attributes, including the specific intent of said attributes.

5

40. A method as defined in claim 36, further including specifying an Event Change object class or classes that specify change event registration for detecting changes in the data objects.

10

41. A method as defined in claim 37, further including specifying a TypeMetaData class of objects that specify semantic information indicating data processing to be available for specified attributes and intended use of the attributes of a data object.

15

42. A method as defined in claim 41, further including specifying a TypeIO class of objects that define a desired format of a data object attribute specified by a TypeMetaData class.

20

43. A method as defined in claim 39, further including specifying an attribute alias by a user to indicate any data source attributes of an attribute domain that may be substituted with attributes of a different attribute domain.

25

44. A method as defined in claim 37, wherein the attribute/metadata interface provides input/output functions for display and editing of the attribute during runtime.



45. A method as defined in claim 37, wherein the attribute/metadata object class is a FieldMetaData attribute that specifies processing criteria for an attribute of a data object.

5 46. A method as defined in claim 37, wherein the processing criteria relates to a selected attribute of the data object for display processing.

10 47. A method as defined in claim 38, wherein the indicated relationship may be resolved using a version of the Relationship object in cache of the computer system.

15 48. A method as defined in claim 36, further including providing a collaboration facility that permits a user at a network computer remote from the computer system to share a view of a data object at the remote network computer.

20 49. A method as defined in claim 48, wherein the collaboration facility comprises a MetaView class of data objects that lightweight encapsulate elements of a data object for visual reconstruction on a display screen of the remote network computer.

50. A method as defined in claim 37, wherein the data interface provides an override function that can override default metadata of the data object attributes.

25 51. A method as defined in claim 39, wherein the data interface provides an override function that can override default metadata of the data object attributes.

52. A method as defined in claim 36, wherein data objects provided through the data interface comprise information that specifies data attributes, relationships, event broadcasting of changes in all registered data consumer objects, metadata for attributes that provide declarative and procedural information and/or input/output functions for display and editing, and related data items, and wherein the data objects can store references without direct access to a related data item.

53. A method as defined in claim 52, wherein the attributes and relationships include contextual metadata, and the events and methods are defined with respect to the contextual metadata.

54. A method as defined in claim 36, wherein the mediation layer further includes object classes that include one or more methods that provide data exposure, and further including a Data Source Interface object class of data objects with methods that automatically determine which data exposure method will be used for data communications between the data source objects and data consumer objects.

55. A method as defined in claim 54, wherein the determined data exposure method comprises data object reflection and introspection.

56. A method as defined in claim 54, wherein the data exposure methods include a data source object method that provides a standard interface, a data source object and a Translator object that maps an alternate data interface to the standard interface, and a data source object that is inspected by the computer system to determine available data fields which are thereby exposed to data consumer objects by the standard interface.

57. A method as defined in claim 36, wherein the data interface of the mediation layer provides a wrapper for the data objects.

58. A method as defined in claim 57, wherein data exchange occurs without providing contextual data characteristics to a receiving data object.

59. A method as defined in claim 58, wherein the contextual data characteristics are supplied by the data object wrapper.

60. A method as defined in claim 36, wherein data exchange occurs without providing contextual data characteristics to a receiving data object.

61. A method as defined in claim 36, further including a plug-in manager object class that integrates service components into the application.

62. A method as defined in claim 61, wherein data service components interact with other components of the application to determine if a service is required and to determine parameters that are to be interchanged.

63. A method as defined in claim 62, wherein the service determination occurs upon introduction of newly loaded components.

64. A method as defined in claim 36 , further including a translator facility that translates information concerning a data object into (1) attributes and metadata that define groups of data source and data consumer class attributes and (2) an object class that specifies at least one domain method for defining groups of data object attributes.

65. A method as defined in claim 57, further including an InfoModel data object class.

5 66. A method as defined in claim 60, wherein said contextual data characteristics are supplied by an InfoModel object.

67. A method as defined in claim 66, wherein the InfoModel object receives a data source object for exposure to the data consumer objects, selects available  
10 attributes and methods of the received data source object for data characterization, and determines if the addition of attributes or methods to the data source object are appropriate.

68. A method as defined in claim 39, wherein the domain is defined using  
15 XML schema.

69. A computer system that supports an object oriented programming environment, the system comprising:

a processor that executes program instructions to provide the object oriented  
20 programming environment; and

data store means for providing program instructions containing specifications for a set of object classes that can be extended using object oriented principles to define an information handling application with data objects comprising a class of data source objects and a class of data consumer objects, and a mediation layer that  
25 defines an interface between the data source objects and the data consumer objects to permit data information exchange between the two data object classes, such that the

class configuration of the data source objects can be specified independently of the class configuration of the data consumer objects.

70. A computer system as defined in claim 69, wherein the mediation layer  
5 defines a data interface that provides an information exchange standard between the data source objects and the data consumer objects.

71. A computer system as defined in claim 70, wherein the data interface  
10 includes at least one attribute/metadata object class that specifies attributes, including metadata, and that provide declarative and procedural information relating to attributes in said data object.

72. A computer system as defined in claim 70, wherein the data interface  
15 includes at least one Relationship object class that specifies relationships between data source class objects.

73. A computer system as defined in claim 70, wherein the data interface  
20 includes at least one Domain Policy object class that specifies a group of related attributes, methods, and semantic information that indicates data processing to be available for the specified attributes, including the specific intent of said attributes.

74. A computer system as defined in claim 70, wherein the data interface  
includes at least one Event Change object class that specifies change event  
registration for detecting changes in the data objects.

75. A program product for use in a computer system that executes program  
instructions recorded in a computer-readable media to perform a method for

information exchange in a computer system that supports an object oriented programming environment and includes access to data storage containing data objects, the program product comprising:

a recordable media; and

5 a program product of computer-readable instructions executable by the computer system to perform a method comprising:

receiving data specifications for a set of object classes that can be extended using object oriented principles to define an information handling application, wherein the extended objects provide an information handling application that can  
10 receive one or more data objects comprising a class of data source objects, and represent the data source objects in accordance with an Information Model class of objects of a mediation layer that defines a data interface between the data source objects and a class of data consumer objects;

wherein the Information Model objects include methods that permit data  
15 communications or information exchange between the two data object classes, such that the class configuration of the data source objects can be specified independently of the class configuration of the data consumer objects.

76. A program product as defined in claim 75, wherein the method further  
20 includes specifying at least one attribute/metadata object class that specifies attributes, including metadata, and that provides declarative and procedural information relating to attributes in said data object.

77. A program product as defined in claim 75, wherein the method further  
25 includes specifying a Relationship object class or classes that specify relationships between data source objects.

78. A program product as defined in claim 75, wherein the method further includes specifying a Domain Policy or Domain object class or classes that specify a group of related attributes, methods, and semantic information that indicates data processing to be available for the specified attributes, including the specific intent of said attributes.

79. A program product as defined in claim 75, further including specifying an Event Change object class or classes that specify change event registration for detecting changes in the data objects.

80. A method of operating a computer system to execute an information handling application in an object oriented programming environment and to receive data from a class of data source objects and provide a class of data consumer objects with data, the method comprising:

selecting available attributes and methods of a data source object for characterization;

determining attributes or methods to be added to the data source object during application execution; and

receiving the data source object and determining whether to expose or hide available attributes and methods of the object from data consumer objects, wherein data communications between the two data object classes is supported, such that the class configuration of the data source objects can be specified independently of the class configuration of the data consumer objects.